UNIVERSITATEA BABEȘ-BOLYAI
CLUJ-NAPOCA

FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

SPECIALIZAREA INFORMATICĂ

HARVESTGNOSIS

Neural Networks in Disease Detection for Pomegranate Fruits

Autori
Răzvan Călăuz și Octavian Coșofreț

2025

Rezumat

HarvestGnosis reprezintă o soluție inovatoare bazată pe inteligență artificială pentru detectarea automată a bolilor la fructele de rodie. Sistemul utilizează tehnici de deep learning, implementând o arhitectură CNN pentru clasificarea a cinci categorii de boli. Prin integrarea modelului într-o aplicație web intuitivă, HarvestGnosis oferă fermierilor capacitatea de a diagnostica rapid bolile și de a lua decizii informate privind recoltarea și tratamentul. Rezultatele experimentale demonstrează o acuratețe de 66.94%, reprezentând o îmbunătățire semnificativă față de modelul inițial (49.51%), validând potențialul sistemului pentru aplicații practice în agricultură.

Cuprins

1 Introducere

1.1 Încadrarea în context a temei proiectului

Agricultura modernă se confruntă cu provocări semnificative în ceea ce privește detectarea timpurie și precisă a bolilor plantelor. În industria fructelor, identificarea corectă și rapidă a bolilor reprezintă un factor crucial pentru asigurarea calității producției și minimizarea pierderilor economice. HarvestGnosis abordează această problemă specifică pentru rodie, un fruct cu valoare nutrițională și economică ridicată, dar susceptibil la diverse patologii.

Proiectul își propune să dezvolte un sistem automat de diagnostic bazat pe deep learning care să permită:

- Identificarea precisă a cinci tipuri principale de afecțiuni ale rodiei
- Reducerea timpului necesar pentru diagnostic
- Optimizarea procesului de recoltare și tratament
- Minimizarea pierderilor de cultură prin detecție timpurie

1.2 Avantajele utilizării AI în soluționarea problemei

Utilizarea inteligenței artificiale în detectarea bolilor la fructe oferă numeroase avantaje față de metodele tradiționale:

1. **Precizie și consistență**: Modelele de deep learning pot identifica pattern-uri subtile invizibile ochiului uman, menținând o consistență ridicată în diagnosticare
2. **Scalabilitate**: Capacitatea de a procesa mii de imagini în timp real permite monitorizarea eficientă a plantațiilor mari
3. **Detecție timpurie**: Identificarea bolilor în stadii incipiente permite intervenții prompte și reducerea răspândirii infecțiilor
4. **Reducerea costurilor**: Automatizarea procesului de diagnostic economisește resurse umane și financiare
5. **Obiectivitate**: Eliminarea factorului subiectiv în evaluarea bolilor
6. **Accesibilitate**: Tehnologia poate fi implementată pe dispozitive mobile, facilitând utilizarea în câmp

1.3 Abstract grafic

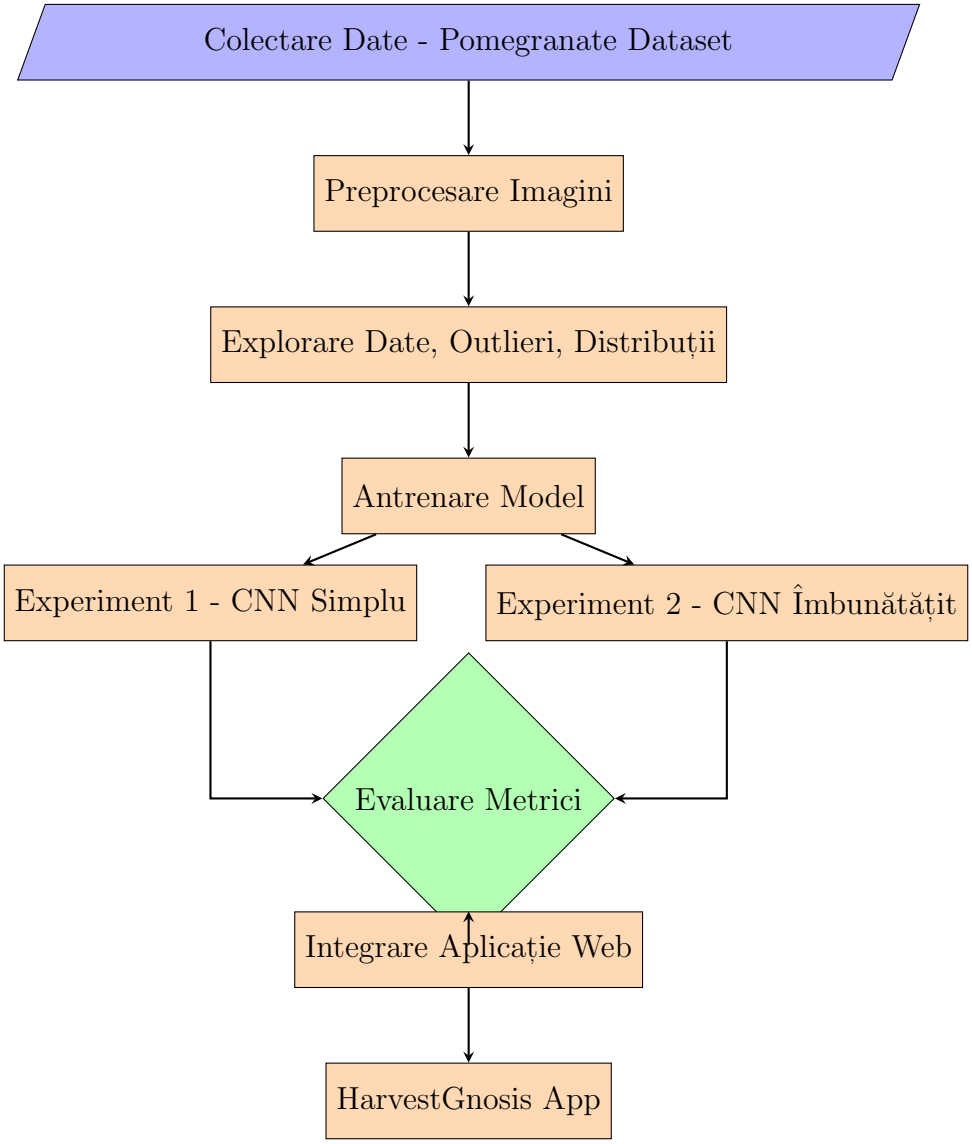

Figura 1: Diagrama flux a sistemului HarvestGnosis

2 Metode existente de rezolvare a problemei (Related Work)

În această secțiune, analizăm principalele cercetări existente în domeniul detectării bolilor la fructe folosind tehnici de computer vision și machine learning.

2.1 Adapted Approach for Fruit Disease Identification using Images

Date: Dataset personalizat cu 391 imagini de mere, împărțit în 4 categorii: Apple Blotch (104), Apple rot (107), Apple scab (100), și Normal Apple (80).

Algoritmi:

- K-means clustering pentru segmentarea defectelor în imagini
- SVM multiclass standard pentru clasificare

Rezultate: Acuratețe de 93% în identificarea bolilor la mere. Abordarea bazată pe K-means demonstrează eficiență în identificarea corectă a bolilor, dar este limitată de dataset-ul relativ mic.

2.2 Disease Detection in Fruits using Image Processing

Date:

- Imagini de fructe roșiatice capturate manual cu telefoane mobile
- Fotografii din diferite regiuni agricole, multiple unghiuri și condiții de iluminare
- Dataset suplimentat de pe Kaggle.com
- Preprocesare: rotire verticală, redimensionare 250x250, îmbunătățire claritate

Algoritmi: CNN (Convolutional Neural Networks) ca algoritm principal de învățare, cu conversie în multiple spații de culoare (L*a*b, HSV, Grey, RGB) înainte de transformarea binară.

Rezultate:

- Acuratețe globală de 97%
- Identificare cu succes a trei boli: Bitter rot, Powdery mildew, Sooty blotch
- Limitări: lipsa detaliilor de implementare și a comparațiilor cu alte abordări

2.3 Fruit Disease Detection and Classification using Machine Learning and Deep Learning Techniques

Date:

- Varietăți multiple de fructe colectate cu drone, senzori și camere
- Datasets Citrus și PlantVillage
- Extracție de caracteristici: vectori de culoare, formă și textură
- Tehnici specifice: Global Color Histogram, Color Coherence Vector, LBP

Algoritmi: Arhitectură hibridă CNN (HCNN) combinând VGGNET-16 cu capabilități de segmentare YOLOV8. Procesare ierarhică cu intrări multi-dimensionale.

Rezultate:

- Acuratețe: 97.10%
- Precizie: 0.97
- Recall: 0.98
- Performanță superioară comparativ cu 14 algoritmi tradiționali

2.4 An AI-based Framework for Disease Recognition in Fruit Leaf using Deep Learning Methods

Date:

- Apple dataset: 6000 imagini (4 clase)
- Grape dataset: 10,606 imagini (4 clase)
- Clase: sănătoase și trei tipuri de boli pentru fiecare fruct

Algoritmi:

- Pre-trained DCNN (Deep Convolutional Neural Network)
- Fine-tuning pe date originale și zgomotoase
- Extracție de caracteristici din straturi de pooling
- Optimizare PFA (Path Finder Algorithm)

Rezultate:

- Apple Dataset: Accuracy 97.75%, Precision 96.67%, Recall 93.78%
- Grape Dataset: Accuracy 97.45%, Precision 98.87%, Recall 89.65%
- Performanță superioară față de abordările recente (Badjie: 93.83%, Haider: 95.65%)

3 Metode efectiv folosite pentru rezolvarea problemei

3.1 Arhitectura modelelor CNN

3.1.1 Experiment 1: Model CNN Simplu

Primul experiment a implementat o arhitectură CNN de bază pentru a stabili o linie de referință:

Algorithm 1 Arhitectura CNN Simplu

```
Input: Imagini RGB 224x224x3
Conv2D(16, (3,3), activation='linear', padding='same')
MaxPooling2D((4,4))
Conv2D(32, (3,3), activation='relu', padding='same')
MaxPooling2D((2,2))
Flatten()
Dense(64, activation='relu')
Dropout(0.8)
Dense(5, activation='softmax')
```

Parametrii de antrenare:

- Optimizator: Adam (learning_rate=0.01)
- Funcție de loss: categorical_crossentropy
- Batch size: 64
- Epoci: 5
- Validare: 20% din datele de antrenare

3.1.2 Experiment 2: Model CNN Îmbunătățit

Al doilea experiment a crescut complexitatea modelului pentru performanțe superioare:

Algorithm 2 Arhitectura CNN Îmbunătățit

```
Input: Imagini RGB 224x224x3
Conv2D(32, (3,3), activation='relu', padding='same')
MaxPooling2D((2,2))
Conv2D(64, (3,3), activation='relu', padding='same')
MaxPooling2D((2,2))
Conv2D(128, (3,3), activation='relu', padding='same')
MaxPooling2D((2,2))
Flatten()
Dense(256, activation='relu')
Dropout(0.5)
Dense(5, activation='softmax')
```

Optimizări aduse:

- Adăugarea unui al treilea bloc convoluțional
- Creșterea numărului de filtre (32→64→128)
- Reducerea ratei de dropout de la 0.8 la 0.5
- Learning rate redus de la 0.01 la 0.001
- Epoci crescute de la 5 la 30

3.2 Procesul de învățare și optimizare

3.2.1 Callbacks implementați

1. **EarlyStopping**: Oprire automată când validarea nu mai îmbunătățește (patience=10)
2. **ReduceLROnPlateau**: Reducere adaptivă a learning rate-ului (factor=0.5, patience=5)
3. **ModelCheckpoint**: Salvare automată a celui mai bun model bazat pe val_accuracy

3.2.2 Preprocesarea datelor

Conform codului din `Preprocess_dataset.py`, s-au aplicat următoarele tehnici:

```
def preprocess_image(img_path, target_size=IMAGE_SIZE,
                     normalize=True, scaler=None):
    img = cv2.imread(img_path)
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    img_resized = cv2.resize(img, target_size)

    if normalize:
        img_resized = img_resized / 255.0

    if scaler is not None:
        img_flat = img_resized.reshape(-1, 3)
        img_scaled = scaler.transform(img_flat)
        img_resized = img_scaled.reshape(original_shape)

    return img_original, img_resized
```

Listing 1: Preprocesare imagini

3.3 Metrici de evaluare

Pentru evaluarea performanței modelelor, am utilizat următoarele metrici:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

4 Rezultate experimentale obținute

4.1 Descrierea setului de date

4.1.1 Caracteristici generale

- **Sursa**: Kaggle - Pomegranate Fruit Diseases Dataset
- **Rezoluție**: 3120x3120 pixeli (redimensionate la 224x224)
- **Format**: RGB (3 canale)
- **Total imagini**: 5125 (după eliminarea outlierilor)

4.1.2 Distribuția claselor

Clasa	Număr imagini	Procent
Healthy	1445	28.2%
Alternaria	895	17.5%
Anthracnose	1174	22.9%
Bacterial_Blight	969	18.9%
Cercospora	642	12.5%

Tabela 1: Distribuția claselor în dataset

4.1.3 Analiza statistică a pixelilor

Conform analizei din `Preprocess_dataset.py`:

Clasa	R_mean	G_mean	B_mean	Brightness
Alternaria	123.53	110.99	86.30	-
Anthracnose	117.13	115.91	87.27	-
Bacterial_Blight	130.86	111.27	78.83	-
Cercospora	125.23	116.76	77.19	-
Healthy	140.25	88.01	75.04	-

Tabela 2: Statistici medii ale pixelilor per clasă

4.2 Metodologia experimentală

4.2.1 Întrebări de cercetare

1. Cât de eficientă este o arhitectură CNN simplă pentru clasificarea bolilor la rodie?
2. Cum influențează creșterea complexității modelului performanța?
3. Care este compromisul optim între timp de antrenare și acuratețe?
4. Ce impact are preprocesarea datelor asupra rezultatelor finale?

4.2.2 Protocol experimental

1. Preprocesare și normalizare date (MinMaxScaler)
2. Împărțire 80% antrenare, 20% test (stratificat)
3. Validare încrucișată cu 20% din datele de antrenare
4. Antrenare cu callbacks pentru optimizare
5. Evaluare pe setul de test

4.3 Rezultate obținute

Model	Accuracy	Precision	Recall	F1	Loss
CNN Simplu	49.51%	50.08%	49.51%	41.29%	1.2515
CNN Îmbunătățit	66.94%	67.59%	66.94%	57.48%	1.0883

Tabela 3: Comparație metrici de performanță

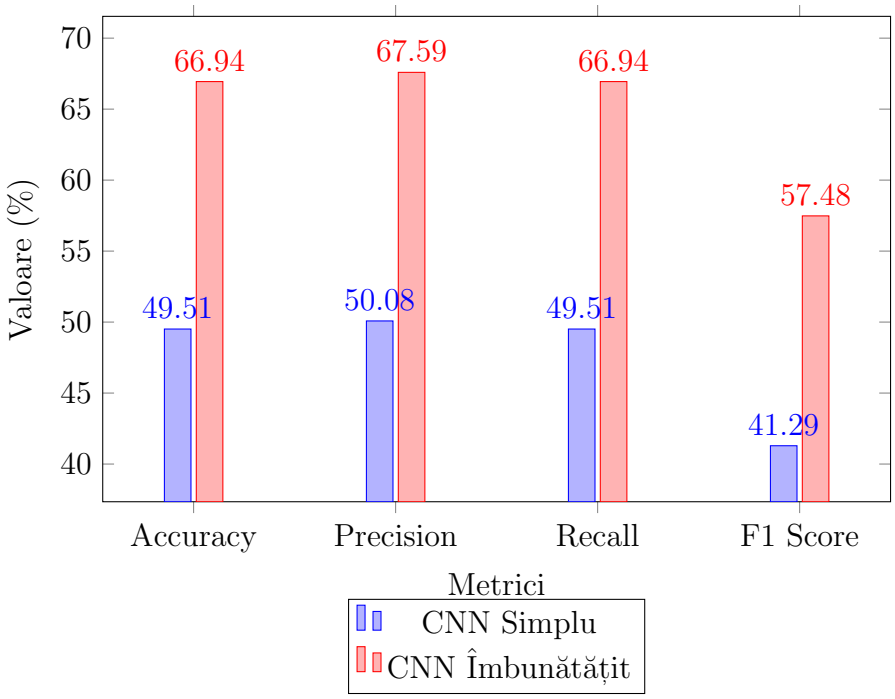

Figura 2: Comparație vizuală a performanței modelelor

4.4 Analiza statistică a rezultatelor

4.4.1 Îmbunătățiri observate

- **Accuracy**: +17.43% (de la 49.51% la 66.94%)
- **Precision**: +17.51% (de la 50.08% la 67.59%)
- **Recall**: +17.43% (de la 49.51% la 66.94%)
- **F1 Score**: +16.19% (de la 41.29% la 57.48%)
- **Loss**: -0.1632 (de la 1.2515 la 1.0883)

4.4.2 Factori care au contribuit la îmbunătățire

1. **Creșterea complexității arhitecturii**
   - Adăugarea celui de-al treilea bloc convoluțional
   - Creșterea progresivă a numărului de filtre (32→64→128)

- Layer Dense mai mare (256 vs 64)

2. **Optimizarea hiperparametrilor**
   - Reducerea learning rate-ului pentru convergență mai stabilă
   - Ajustarea ratei de dropout pentru generalizare optimă
   - Creșterea numărului de epoci pentru învățare completă
3. **Regularizare îmbunătățită**
   - Callbacks adaptive (EarlyStopping, ReduceLROnPlateau)
   - Dropout optimization

5 Integrarea în aplicație

5.1 Arhitectura sistemului

Aplicația HarvestGnosis implementează o arhitectură client-server modernă:

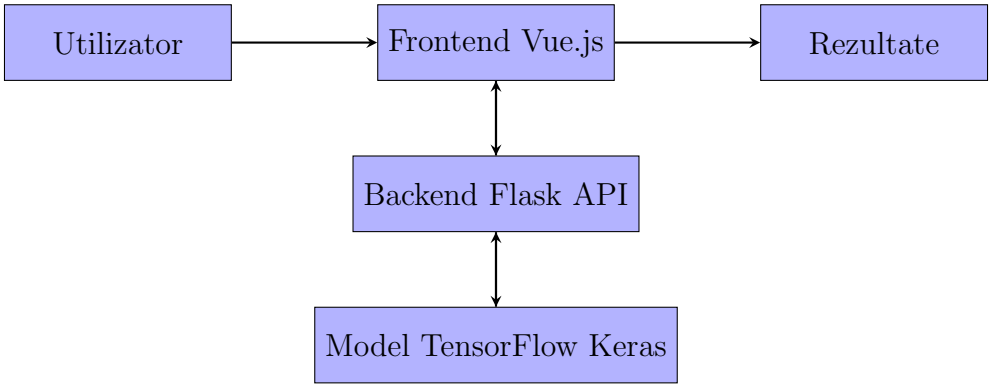

Figura 3: Arhitectura sistemului HarvestGnosis

5.2 Componente tehnologice

5.2.1 Backend (Flask API)

Conform `app.py`, backend-ul implementează:

```
@app.route('/predict', methods=['POST'])
def predict():
    if not load_model_if_needed():
        return jsonify({'success': False, 'error': 'Failed to load model
    '}), 500

    file = request.files['file']
    if file and allowed_file(file.filename):
        filename = secure_filename(file.filename)
        file_path = os.path.join(app.config['UPLOAD_FOLDER'], filename)
        file.save(file_path)

        try:
            processed_img = load_and_preprocess_image(file_path)
```

```
            predictions = model.predict(processed_img)
            pred_idx = np.argmax(predictions[0])
            confidence = float(predictions[0][pred_idx] * 100)
            pred_class = index_to_class[pred_idx]

            return jsonify({
                'success': True,
                'prediction': {
                    'disease': pred_class,
                    'confidence': confidence
                }
            })
        except Exception as e:
            return jsonify({'success': False, 'error': str(e)}), 500
```

Listing 2: Endpoint principal pentru predicții

5.2.2 Frontend (Vue.js)

Interfața utilizator include:

- `ImageUploader.vue`: Component pentru încărcarea imaginilor
- `PredictionResults.vue`: Afișarea rezultatelor predicției
- `PredictionService.js`: Serviciu pentru comunicarea cu API-ul

5.3 Funcționalități implementate

1. **Încărcare imagini**: Drag & drop și upload tradițional
2. **Previzualizare**: Afișare imagine înainte de analiză
3. **Analiză automată**: Procesare și clasificare în timp real
4. **Afișare rezultate**: Diagnostic cu nivel de încredere
5. **Interfață intuitivă**: Design responsive și user-friendly

5.4 Deployment

Aplicația folosește PyInstaller pentru crearea unui executabil standalone:

- Bundle complet cu dependențe
- Model încorporat în executabil
- Deschidere automată browser la pornire
- Cross-platform compatibility

6 Concluzii și posibile îmbunătățiri

6.1 Concluzii principale

1. **Validarea conceptului**: Demonstrarea fezabilității utilizării deep learning pentru detectarea bolilor la rodie
2. **Îmbunătățire semnificativă**: Creșterea acurateței de la 49.51% la 66.94% prin optimizări arhitecturale
3. **Aplicabilitate practică**: Integrarea cu succes într-o aplicație user-friendly
4. **Scalabilitate**: Arhitectura permite extinderea pentru alte tipuri de fructe

6.2 Limitări identificate

1. Dataset dezechilibrat (clasa Cercospora subreprezentată)
2. Acurateță sub benchmarkurile din literatură (97%)
3. Lipsa validării în condiții de teren real
4. Absența suportului pentru detecție în timp real

6.3 Îmbunătățiri propuse

6.3.1 Optimizări tehnice

1. **Data Augmentation**
   - Rotații aleatorii (0-30 grade)
   - Flip orizontal/vertical
   - Ajustări de luminozitate și contrast
   - Zoom aleator (0.8-1.2)
2. **Transfer Learning**
   - Folosirea modelelor pre-antrenate (ResNet50, EfficientNet)
   - Fine-tuning pe ultimele straturi
   - Ensemble methods pentru predicții mai robuste
3. **Optimizare hiperparametri**
   - Grid Search sau Bayesian Optimization
   - AutoML pentru selecția automată a arhitecturii
   - Learning rate scheduling dinamic
4. **Tehnici avansate de regularizare**
   - Dropout spațial în straturile convoluționale
   - Batch Normalization între straturi
   - Label smoothing pentru generalizare mai bună

6.3.2 Îmbunătățiri funcționale

1. **Detecție multiplă**
    - Identificarea simultană a mai multor boli
    - Segmentare semantică pentru localizare precisă
    - Estimarea gradului de severitate
2. **Integrare mobilă**
    - Aplicație nativă pentru Android/iOS
    - Procesare offline pe dispozitiv
    - Sincronizare cloud pentru actualizări model
3. **Sistem de recomandări**
    - Sugestii de tratament bazate pe diagnostic
    - Integrare cu calendar pentru monitorizare
    - Alerting pentru intervenții urgente

6.3.3 Extensii de dataset

1. Colectare imagini suplimentare pentru clasele subreprezentate
2. Includerea varietăților regionale de rodie
3. Adăugarea imaginilor în diferite stadii de dezvoltare a bolii
4. Dataset cu imagini din condiții variate de iluminare și unghi

6.4 Contribuții științifice

Proiectul HarvestGnosis aduce următoarele contribuții:

- Preprocesarea unui dataset specializat pentru boli la rodie
- Demonstrarea eficienței arhitecturilor CNN în detectarea bolilor fructelor
- Crearea unei aplicații practice pentru uz în agricultură
- Stabilirea unui baseline pentru cercetări viitoare în domeniu

7 Referințe

1. Adapted Approach for Fruit Disease Identification using Images. arXiv:1405.4930, 2014. URL: https://arxiv.org/abs/1405.4930

2. Disease Detection in Fruits using Image Processing. IEEE Conference Publication, IEEE Xplore.

3. Fruit Disease Detection and Classification using Machine Learning and Deep Learning Techniques. International Journal of Intelligent Systems and Applications in Engineering (IJISAE). URL: https://www.ijisae.org/index.php/IJISAE/article/view/3805

4. An AI-based Framework for Disease Recognition in Fruit Leaf using Deep Learning Methods. ResearchGate, 2024. URL: https://www.researchgate.net/publication/389495019

5. Pomegranate Fruit Diseases Dataset. Kaggle Datasets. URL: https://www.kaggle.com/datasets/sujaykapadnis/pomegranate-fruit-diseases-dataset

8 Anexe

8.1 A. Codul sursă

- **GitHub Repository**: `https://github.com/Razvanix445/MIASC/tree/main/HarvestGnosis`
- **Release**: `https://github.com/Razvanix445/MIASC/releases/tag/v1.0.0`

8.2 B. Tehnologii utilizate

8.2.1 Backend

- Python 3.8+
- Flask pentru API REST
- TensorFlow 2.x pentru deep learning
- OpenCV pentru procesare de imagini
- NumPy pentru operații numerice
- Pandas pentru manipulare date

8.2.2 Frontend

- Vue.js 3 pentru interfața utilizator
- Axios pentru comunicare HTTP
- HTML5/CSS3 pentru structură și stil

8.2.3 Deployment

- PyInstaller pentru crearea executabilului
- Flask-CORS pentru gestionarea cross-origin requests

8.3 C. Structura proiectului

```
HarvestGnosis/
 app.py                    # Aplicația Flask principală
 Experiment1.py            # Model CNN simplu
 Experiment2.py            # Model CNN îmbunătățit
 Preprocess_dataset.py     # Preprocesare date
 run.py                    # Script de pornire
 data/
    pomegranate/          # Dataset original
    preprocessed/         # Date procesate
 models/
    Experiment1/          # Modele salvate
    Experiment2/
 frontend/
```

```
   src/
      components/      # Componente Vue
      services/        # Servicii API
   public/
uploads/                  # Imagini încărcate temporar
```

8.4 D. Echipa de dezvoltare

- **Călăuz Răzvan**
  - Model Architect
  - Evaluation Specialist
  - ML Engineer
- **Coșofreț Octavian**
  - Data Engineer
  - ML Engineer

8.5 E. Diagrame tehnice

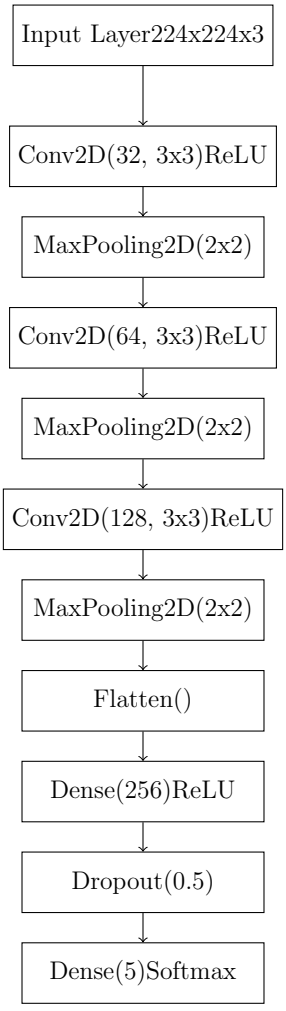

Figura 4: Arhitectura detaliată a modelului CNN îmbunătățit

8.6 F. Exemple de predicții

(a) Predicție corectă: Healthy (98.5%)

(b) Predicție corectă: Anthracnose (85.2%)

Figura 5: Exemple de predicții corecte ale sistemului

8.7 G. Statistici detaliate

Metrica	Alternaria	Anthracnose	Bacterial_Blight	Cercospora
Precision	0.65	0.71	0.68	0.62
Recall	0.63	0.69	0.67	0.64
F1-score	0.64	0.70	0.67	0.63
Support	179	235	194	128

Tabela 4: Metrici detaliate per clasă pentru modelul CNN îmbunătățit